Process for Validating Assignments

Computational Physics - Assignments

Dr Kieran Murphy

SETU (Waterford)

Autumn Semester, 2026/27

Introduction

- Motivation and overview of the assignment validation process.

Motivation and Overview

Traditionally the grading of assignments was more focused on whether the submitted work met the stated requirements. However, now in the presence of AI, we need to be more interested in the demonstration of understanding of the submitted work, i.e., assuming that the work is correct, how can we validate that the author has a good understanding of the underlying concepts?

- Assignments are submitted and graded as normal.
- You are then invited to a zoom session where you share your screen and explain your work. You are asked questions about your work, including questions about the underlying concepts and issues related to the assignment. You may also be asked to demonstrate core scientific computing skills and show that you can modify your implementation to meet new requirements.

We will run a demo zoom session during the semester so you can get a feel for what the validation process will involve.

Structure of the Validation Process

- Individual zoom sessions, student sharing their screen and explaining their work.
- Trial (formative) session, about week 6, focusing on the probability task assignments.
- Final session, about week 15, covering entire module, to generate a single `validation_score` for the module.
- Questions covering core tasks and concepts/issues related to assignments.

Little or no understanding, unable to perform core tasks	Limited understanding, only able to perform the basic core tasks	Understand assignments but can't modify them, capable of performing some core tasks	Understand assignments, can modify them, able to perform some core tasks	Clearly understand assignments, can modify them, able to perform core tasks
< 40%	40%–49%	50%–59%	60%–79%	80%–100%

- Final grade for each assignment is computed using

$$(\text{individual_assignment_mark}) \times (\text{validation_score})$$

- Can use AI and other resources during zoom session but be mindful that it is your understanding that is under review, not an AI's ability to generate.

Typical Scientific Python Tasks

- Demonstrate the ability to implement typical computational physics tasks using scientific python (`numpy`/`scipy`/`matplotlib`, ...)
- Within each task there is a wide range of complexity, but underlying concepts are the same.

Define a Function

Define a function using `def`, and an anonymous function using `lambda`.

- Complexity of function can vary. We have functions …

… direct python implementation	… requiring use of `numpy`	… requiring use of `scipy`	… defined in terms of other functions
$f(x) = x^2 + 1$	$f(x) = \cos(x)$ $f(x) = e^{-x}\sin(x)$	$f(x) = \int_0^x e^{-t^2} dt$ `scipy.special` functions	$f(\alpha, \beta) = \max_t I(t; \alpha, \beta)$ where $I(t; \alpha, \beta, t)$ is number of infected individuals at time t in a SIR model with parameters α and β

- Implement function and write code to verify its implementation.

Graphing

Given a function, plot its graph over a specified input range. Use appropriate labels and titles for the graph.

- Multiple variations on core task:
  - Plotting tabular data.
  - Examine parameter-dependent functions on the same graph, e.g., see effect of α in $f(x;\alpha) = x^{\alpha}$, or $f(x;A,\omega) = A\cos(\omega x)$, ...
  - Plotting multiple functions on same graph.
- More advanced variations:
  - Using subplot (and sharing axes) to link related graphs.
  - Adding annotations and legends to the graphs.
  - Customizing the appearance of the graphs (colors, line styles, etc.).

Root Finding

Given a function $f(x)$, determine the root of $f(x)$ using numerical methods from `scipy`

- Simplest case is a continuous function with a unique root.
- More complex cases include:
  - Multiple roots.
    - So you need to be careful about the initial guess for the root-finding algorithm or bracket the root.
  - Discontinuous functions or functions with singularities.
    - So you need to bracket the root to avoid the discontinuity/singularity.
  - **Parameter-dependent functions.**
    - So you need to pass the parameters to the root-finding function.
- ~~Fixed point methods~~
  - ~~Convert a root-finding problem into a fixed-point problem, e.g.,~~ $f(x) = 0$ ~~can be rewritten as~~ $x = g(x)$ ~~for some function~~ $g(x)$~~.~~

Integration

Given a function $f(x)$, compute the definite integral of $f(x)$ over a specified interval using numerical methods from `scipy`.

- Simplest case is an integrand (function to be integrated) of a single variable over a finite interval, but more complex cases include:
  - Multiple dimension integrals.
  - Infinite intervals of integration.
  - Functions defined in terms of other functions.
  - Functions with singularities.
  - Parameter-dependent functions.

Differentiation

Given a function compute its derivative using methods from `numpy`/`scipy` ~~(and possibly `numdifftools`)~~.

- Simplest case is a scalar function of a single variable, but more complex cases include:
  - Functions of multiple variables.
  - Functions defined in terms of other functions.
  - Parameter-dependent functions.
- Able to show accuracy of the computed derivative by comparing with analytical derivative (if known) ~~or using Richardson extrapolation.~~

Integration of ODE

Given an ordinary differential equation (ODE) use numerical methods from `scipy` to solve the ODE over a specified interval and satisfy initial and/or boundary conditions.

- Simplest case is a scalar, first-order ODE of the form $\frac{dy}{dt} = f(t, y)$ with an initial condition $y(t_0) = y_0$.
- More complex variations include:
  - Systems of ODEs.
  - Higher-order ODEs
    - You need to convert to a system of first-order ODEs.
  - ODEs with boundary conditions.
    - You need to use a shooting method or use root finding with the standard initial value problem approach.

Curve Fitting

Given a set of data points, fit a curve to the data using numerical methods from `scipy` and determine the parameters of the fitted curve.

- Read and parse data from a file (typically CSV) using `numpy`/`pandas`.
- Verify the quality of the fit using visual and statistical measures (e.g., R-squared, residuals, etc.).
- Simplest case is fitting a single parameter function to 2D data, but more complex variations include:
  - Functions with multiple parameters.
  - Fitting curves with constraints on the parameters.
  - Fitting curves to data with outliers.
  - Fitting multiple curves to the same data to determine best functional form.

Linear Algebra

Given a system of linear equations, use numerical methods from `numpy`/`scipy` to solve the system and determine the solution vector.

- Define matrices and vectors using `numpy` arrays, and apply linear algebra operations such as matrix multiplication, inversion, and transposition.
- Solve eigenvalue/eigenvector problems.

Sample Assignments Questions

- Sample questions related to assignments (assignments change from year to year, but core concepts remain the same).
- Demonstrate understanding beyond simple reading of code.
- Demonstrate ability to modify implementation to meet new requirements.
- Demonstrate understanding of underlying problem and its behaviour as parameters change.

Example: The Nearly Intoxicated Waiter

Using the Nearly Intoxicated Waiter Problem as example context:

- Typical questions related to this assignments include:
  - How does accuracy of simulation depend on the number of customers?
  - What is the limiting behaviour as the number of customers increase?
- Typical modifications to the assignment include:
  - Change the number of people who get an incorrect order.
  - Plot change in probability as number of customers getting correct order increases.
  - Presume the customers are sitting at a circular table, and customers are willing to swap their orders with their neighbours. So we can treat a correct order as being placed at the correct position or at one of the two neighbouring positions. How does this affect the probability that exactly one person gets their correct order?
  - Say the waiter is given a second chance to correct all incorrect orders, so they take back only the dishes of people who got an incorrect order, and serve them out again at random (they are still drunk). How does this affect the probability that exactly one person gets their correct order?